IFSB-ReliefF: یک روش انتخاب نمونه و ویژگی هم‌زمان بر مبنای ReliefF

زینب عباسی، محسن رحمانی* و حسین غفاریان
گروه کامپیوتر، دانشکده فنی و مهندسی، دانشگاه اراک، اراک، ایران

چکیده

افزایش استفاده از اینترنت و برخی از پدیده‌ها مانند شبکه‌های حس‌گر، منجر به افزایش غیرضروری اطلاعات شده است. اگرچه این امر مزایای بسیاری دارد، اما باعث ایجاد مشکلاتی مانند نیاز به فضای ذخیره‌سازی و پردازنده‌های بهتر و همچنین پالایش اطلاعات برای حذف اطلاعات غیرضروری می‌شود. الگوریتم‌های کاهش داده، روش‌هایی برای انتخاب اطلاعات مفید از مقدار زیادی داده‌های تکراری، ناقص و زائد فراهم می‌کنند. در این مقاله، الگوریتم ReliefF که یک الگوریتم رتبه‌بندی ویژگی است، تغییر داده شده تا به‌طور هم‌زمان ویژگی‌ها و نمونه‌ها را انتخاب کند. الگوریتم پیشنهادشده می‌تواند بر روی ویژگی‌های اسمی و عددی و مجموعه‌داده‌ها با مقادیر مفقود اجرا و هم‌چنین، می‌تواند به‌صورت موازی روی یک پردازندهٔ چندهسته‌ای اجرا شود، که این امر باعث کاهش بسیار چشم‌گیر زمان اجرا و امکان اجرای آن روی مجموعه‌داده‌های بزرگ می‌شود؛ علاوه‌براین، در این الگوریتم، انتخاب نمونه از هر رده متناسب با احتمال پیشین رده است و در نتیجه توازن و نسبت اولیه رده‌ها در مجموعه اصلی از بین نخواهد رفت. نتایج آزمایش بر روی چهار مجموعه‌داده نشان‌دهندهٔ موفقیت الگوریتم پیشنهادی در این امر است.

واژگان کلیدی: کاهش داده‌ها، انتخاب نمونه، انتخاب ویژگی، ReliefF.

IFSB-ReliefF: A New Instance and Feature Selection Algorithm Based on ReliefF

Zeinab Abbasi, Mohsen Rahmani* & Hossein Ghafarian
Computer Department, faculty of engineering, Arak University, Arak, Iran

Abstract

Increasing the use of Internet and some phenomena such as sensor networks has led to an unnecessary increasing the volume of information. Though it has many benefits, it causes problems such as storage space requirements and better processors, as well as data refinement to remove unnecessary data. Data reduction methods provide ways to select useful data from a large amount of duplicate, incomplete and redundant data. These methods are often applied in the pre-processing phase of machine learning algorithms. Three types of data reduction methods can be applied to data: 1. Feature reduction.2. Instance reduction: 3. Discretizing feature values.

In this paper, a new algorithm, based on ReliefF, is introduced to decrease both instances and features. The proposed algorithm can run on nominal and numeric features and on data sets with missing values. In addition, in this algorithm, the selection of instances from each class is proportional to the prior probability of classes. The proposed algorithm can run parallel on a multi-core CPU, which decreases the runtime significantly and has the ability to run on big data sets.

One type of instance reduction is instance selection. There are many issues in designing instance selection algorithms such as representing the reduced set, how to make a subset of instances, choosing

* Corresponding author
*نویسندهٔ عهده‌دار مکاتبات

• تاریخ ارسال مقاله: ۱۳۹۷/۰۷/۰۵ • تاریخ پذیرش: ۱۳۹۸/۰۶/۱۱ • تاریخ انتشار: ۱۳۹۹/۱۲/۰۴ • نوع مطالعه: پژوهشی

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

distance function, evaluating designed reduction algorithm, the size of reduced data set and determining the critical and border instances. There are three ways of creating a subset of instances. 1) Incremental. 2) Decremental. 3) Batch. In this paper, we use the batch way for selecting instances. Another important issue is measuring the similarity of instances by a distance function. We use Jaccard index and Manhattan distance for measuring. Also, the decision on how many and what kind of instances should be removed and which must remain is another important issue. The goal of this paper is reducing the size of the stored set of instances while maintaining the quality of dataset. So, we remove very similar and non-border instances in terms of the specified reduction rate.

The other type of data reduction that is performed in our algorithm is feature selection. Feature selection methods divide into three categories: wrapper methods, filter methods, and hybrid methods. Many feature selection algorithms are introduced. According to many parameters, these algorithms are divided into different categories; For example, based on the search type for the optimal subset of the features, they can be categorized into three categories: Exponential Search, Sequential Search, and Random Search. Also, an assessment of a feature or a subset of features is done to measure its usefulness and relevance by the evaluation measures that are categorized into various metrics such as distance, accuracy, consistency, information, etc.

ReliefF is a feature selection algorithm used for calculating a weight for each feature and ranking features. But this paper is used ReliefF for ranking instances and features. This algorithm works as follows: First, the nearest neighbors of each instances are found. Then, based on the evaluation function, for each instance and feature, a weight is calculated, and eventually, the features and instances that are more weighed are retained and the rest are eliminated. IFSB-ReliefF (Instance and Feature Selection Based on ReliefF) algorithm is tested on two datasets and then C4.5 algorithm classifies the reduced data. Finally, the obtained results from the classification of reduced data sets are compared with the results of some instance and feature selection algorithms that are run separately.

Keywords: data reduction, instance selection, feature selection, ReliefF.

۱- مقدمه

در سال‌های اخیر، به‌دلیل استفاده از اینترنت و به‌خصوص شبکه‌های اجتماعی، رشد داده‌ها بسیار سریع بوده است. علاوه‌براین، بعضی از ابزارها مانند حس‌گرها مقدار زیادی داده تولید می‌کنند؛ بنابراین ذخیره‌سازی و پردازش نیازمند دستگاه‌های ذخیره‌سازی ویژه و رایانه های پرقدرت است. یکی از چالش‌های ذخیره و پردازش داده‌های بزرگ، فشرده‌سازی و استخراج اطلاعات مفید است. بسیاری از الگوریتم‌ها و روش‌ها برای حل این مشکل ارائه شده‌اند. روش‌های کاهش داده تلاش می‌کنند داده‌های زائد و بی‌ربط را حذف کنند تا دقت نتایج را افزایش داده و یا زمان و هزینه فرآیندها را کاهش دهند [1] . این روش‌ها اغلب در مرحله پیش‌پردازش الگوریتم‌های یادگیری ماشین استفاده می‌شوند.

سه نوع روش کاهش داده می‌تواند بر روی داده‌ها اعمال شود:

۱. کاهش ویژگی‌ها: این روش‌ها سعی می‌کنند ویژگی‌های مربوط را پیدا کرده و ویژگی‌های نوفه‌دار و غیر مفید را حذف کنند. آنها به دو دسته انتخاب ویژگی و استخراج ویژگی تقسیم می‌شوند. روش‌های انتخاب ویژگی، زیرمجموعه‌ای از ویژگی‌ها را انتخاب کرده و بقیه را حذف می‌کنند [2] . روش‌های استخراج ویژگی، ویژگی‌ها را ترکیب کرده و ویژگی‌های جدید تولید می‌کنند. برای مطالعه بیشتر، به [3, 4, 5] مراجعه شود.

۲. کاهش نمونه: مانند کاهش ویژگی، دو نوع روش وجود دارند که عبارتند از: کاهش نمونه (یا رکورد)، که نمونه‌برداری (یا انتخاب نمونه) نامیده می‌شود و فشرده‌سازی اطلاعات. در نوع نخست نمونه‌ها با روش‌های تصادفی [6] یا روش‌های خاص [7] انتخاب شده و بقیه حذف می‌شوند. مانند روش‌های استخراج ویژگی، در فشرده‌سازی اطلاعات نمونه‌ها با هم ترکیب شده و به شکل محدودتر و فشرده تبدیل می‌شوند. برای مطالعه بیشتر به [8, 9, 10, 11] مراجعه شود.

۳. جداسازی مقادیر ویژگی‌ها: این روش‌ها دامنه مقادیر برای هر ویژگی را با گسسته‌کردن مقادیر ویژگی‌های عددی پیوسته کاهش می‌دهند [12] .

همان‌طور که درقبل گفته شد، یک نوع از روش‌های کاهش داده، کاهش نمونه است. از آنجایی که مجموعه‌داده‌های حجیم، فراتر از قابلیت ذخیره و پردازش سامانه‌های معمولی هستند [13] ، این نوع کاهش داده لازم است. یک نوع از کاهش نمونه، انتخاب نمونه‌ها است. دو دلیل برای افزایش دقت الگوریتم‌های یادگیری ماشین با انتخاب نمونه وجود دارد. یکی این‌که انتخاب نمونه می‌تواند نوفهٔ در داده‌ها را از بین ببرد. و دیگر این‌که، برخی از نمونه‌ها زائد هستند و انتخاب نمونه به حذف آن‌ها کمک

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

می‌کند. بسیاری از الگوریتم‌های انتخاب نمونه به‌دلیل کارایی کم، کمبود منابع و تعمیم‌ناپذیری نمی‌توانند از عهده مجموعه‌داده‌های بزرگ برآیند[12].

مسائل بسیاری در طراحی الگوریتم‌های انتخاب نمونه وجود دارند، مانند چگونگی نمایش مجموعه کاهش‌یافته، نحوه ایجاد یک زیرمجموعه از نمون‌ها، انتخاب تابع فاصله مناسب، ارزیابی عملکرد الگوریتم انتخاب نمونه طراحی‌شده [14]، شرایط حذف یا نگهداری نمونه‌ها (اندازه مجموعه‌داده‌های کاهش داده شده) و تعیین نمونه‌های بحرانی و نقاط مرزی [15].

سه راه برای ایجاد زیرمجموعه‌ای از نمونه‌ها وجود دارد: ۱) افزایشی[1]: در ابتدا، زیرمجموعه خالی است و سپس بعضی از نمونه‌ها به آن افزوده می‌شوند. ۲) کاهشی[2]: ابتدا تمام مجموعه‌داده به‌عنوان زیرمجموعه نهایی انتخاب‌شده و سپس برخی از نمونه‌ها از آن حذف می‌شوند. ۳) دسته‌ای[3]: ابتدا تمام نمونه‌ها مورد تجزیه و تحلیل قرار می‌گیرند و برای هر یک از آنها یک وزن یا رتبه محاسبه و سپس تمام نمونه‌ها با وزن کم به یک باره حذف می‌شود [16]. در این مقاله، از روش دسته‌ای برای انتخاب نمونه‌ها استفاده می‌شود. یکی دیگر از مسائل مهم، اندازه‌گیری شباهت نمونه‌ها با یک تابع فاصله است. در روش پیشنهادی از شاخص ژاکارد[4] [17] و فاصله منهتن[5] برای اندازه‌گیری شباهت نمونه‌ها استفاده می‌شود. همچنین، تصمیم‌گیری درباره تعداد نمونه‌هایی که باید حذف شوند یا باقی بمانند، یکی دیگر از مسائل مهم است. هدف این مقاله کاهش حجم نمونه‌های ذخیره‌شده در عین حفظ کیفیت مجموعه‌داده‌ها است؛ بنابراین، موارد بسیار مشابه و نقاط غیرمرزی بر حسب نرخ کاهش مشخص‌شده حذف می‌شود.

نوع دیگری از کاهش داده‌ها که در الگوریتم پیشنهادی انجام می‌شود، انتخاب ویژگی است. روش‌های انتخاب ویژگی به سه دسته تقسیم می‌شود: روش‌های بسته‌بندی[6]، روش‌های فیلتر و روش‌های ترکیبی. روش‌های بسته‌بندی از یک الگوریتم داده‌کاوی برای ارزیابی ویژگی‌ها استفاده می‌کنند [18]. این روش‌ها مفید است؛ اما در هنگام استفاده برای داده‌های حجیم کارایی چندانی ندارد [19]. روش‌های فیلتر از الگوریتم‌های داده‌کاوی استفاده نمی‌کنند و از طریق یک تابع ارزیابی، ویژگی‌های مربوطه را جستجو می‌کنند [20]. روش‌های فیلتر سریع و بی‌طرفانه هستند [2]. همچنین روش‌های فیلتر نسبت به روش‌های بسته‌بندی در برخورد با مجموعه‌داده‌های حجیم به‌دلیل انجام محاسبات سریع‌تر ترجیح داده می‌شود. روش‌های ترکیبی دو روش یادشده را ترکیب و از مزایای آنها استفاده می‌کنند [2]. الگوریتم‌های انتخاب ویژگی با توجه به پارامترهای مختلف به دسته‌های مختلف تقسیم می‌شوند. به‌عنوان مثال، بر اساس نوع جستجو برای زیرمجموعهٔ بهینه از ویژگی‌ها، می‌توان آنها را به سه دسته تقسیم کرد: جستجوی نمایی[7](جستجو برای هر زیرمجموعه ممکن برای یافتن زیرمجموعه مطلوب)، جستجوی ترتیبی[8] (اضافه یا حذف ویژگی به مطلوب زیرمجموعه در هر مرحله) و جستجوی تصادفی (انتخاب تصادفی یک زیرمجموعه و ارزیابی آن)[4]. هم‌چنین، ارزیابی یک ویژگی یا زیرمجموعه‌ای از ویژگی‌ها برای سنجش سودمندی و اهمیت آن‌ها با معیارهای مختلف مانند فاصله، دقت، صحت، اطلاعات، و ... اندازه‌گیری می‌شود [2, 4]. نتایج الگوریتم‌های انتخاب ویژگی می‌توانند از یک الگوریتم داده‌کاوی برای مقایسه دقت مجموعه انتخاب شده و مجموعه داده‌های کامل مورد استفاده قرار گیرند [2].

ReliefF یک الگوریتم انتخاب ویژگی است [21]. این الگوریتم برای محاسبه وزن برای هر ویژگی و رتبه‌بندی ویژگی‌ها استفاده، اما در این مقاله، ReliefF برای رتبه‌بندی و انتخاب نمونه‌ها و ویژگی‌ها استفاده می‌شود. الگوریتم پیشنهادشده، ویژگی‌ها و نمونه‌ها را به‌طور هم‌زمان کاهش می‌دهد. هم‌چنین این الگوریتم به‌گونه‌ای طراحی شده است که بتواند با استفاده از با برنامه‌نویسی موازی، هم‌زمان روی نمونه‌ها کار انجام دهد و با اجرای آن روی یک پردازنده چندهسته‌ای، زمان اجرا تا حد بسیار زیادی کاهش می‌یابد. الگوریتم IFSB-ReliefF (انتخاب نمونه و ویژگی بر اساس ReliefF[9]) بر روی چهار مجموعه داده آزمایش شده است و سپس الگوریتم C4.5 [22] داده‌های کاهش‌یافته را رده‌بندی می‌کند؛ درنهایت، نتایج به‌دست‌آمده از رده‌بندی مجموعه‌داده‌های کاهش داده‌شده با نتایج برخی از الگوریتم‌های انتخاب نمونه و ویژگی که به‌طور جداگانه اجرا می‌شوند، مقایسه می‌شود.

مقاله بدین‌صورت سازمان‌دهی شده است: بخش ۲ مقدمات لازم را توضیح می‌دهد و کارهای پیشین را بررسی می‌کند. در بخش ۳، الگوریتم IFSB-ReliefF که برای

[1] Incremental
[2] Decremental
[3] Batch
[4] Jaccard index
[5] Manhattan distance
[6] Wrapper
[7] Exponential
[8] Sequential
[9] IFSB-ReliefF: Instance and Feature Selection Based on ReliefF

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]
[DOI: 10.29252/jsdp.17.4.49]

انتخاب ویژگی و نمونه از ReliefF استفاده می‌کند، ارائه می‌شود. بخش ۴ نتایج آزمایش‌ها و سرانجام در بخش ۵ خلاصه و نتیجه کار ارائه می‌شود.

۲- کارهای مرتبط

در این بخش، ابتدا، الگوریتم ReliefF توضیح داده شده است. برخی از الگوریتم‌های معرفی‌شده هم نمونه و هم ویژگی‌ها را انتخاب می‌کنند. برخی از مقالات در مورد این نوع الگوریتم‌ها در این بخش مورد بررسی قرار گرفته است.

۲-۱- ReliefF

یک راه برای کاهش حجم داده‌ها انتخاب ویژگی [2] است. الگوریتم‌های انتخاب ویژگی می‌توانند به حذف ویژگی‌های زائد و یا دارای نوفه کمک کنند. همچنین می‌توانند زمان اجرا را کاهش داده و دقت الگوریتم‌های داده‌کاوی را افزایش دهند.

Relief [23] یک روش انتخاب ویژگی فیلتری است که از یادگیری مبتنی بر نمونه الهام گرفته شده است. این الگوریتم یک روش پیش پردازش معروف است که می‌تواند در بسیاری از مسائل داده‌کاوی مورد استفاده قرار گیرد [24] . Relief ویژگی‌ها را بر اساس کیفیت آن‌ها به‌طور مؤثر رتبه‌بندی می‌کند. این الگوریتم می‌تواند بر روی مجموعه‌داده‌های اسمی و عددی کار کند [21] . Relief درجه اهمیت ویژگی‌ها را با محاسبه تفاوت بین ویژگی‌ها تخمین می‌زند [24] ؛ اما Relief برخلاف همه محاسن، معایبی نیز دارد. این الگوریتم نمی‌تواند در مجموعه‌داده‌های چندرده‌ای و مجموعه‌داده‌ها با مقادیر مفقود کار کند [21] . به‌منظور غلبه بر این مشکلات، الگوریتم Relief اصلاح‌شده است و الگوریتم جدیدی به نام ReliefF ارائه شد [21] . ReliefF می‌تواند بر روی مجموعه‌داده‌ها با مقادیر مفقود و مجموعه‌داده‌ها با بیش از دو رده داده کار کند. به جای انتخاب یکی از نزدیک‌ترین همسایه‌ها، که در الگوریتم Relief انجام می‌شود، ReliefF تعدادی از نزدیک‌ترین همسایگان یک نمونه انتخابی را می‌یابد؛ علاوه‌براین، ReliefF از یک تابع متفاوت برای محاسبه وزن ویژگی‌ها استفاده می‌کند. این کار برای مدیریت مجموعه‌داده‌های ناقص انجام می‌شود. همان‌طورکه در شکل (۱) نشان داده شده است، الگوریتم ReliefF به‌شرح زیر عمل می‌کند:

در خط ۵، pc احتمال پیشین رده است. در خط 6.a الگوریتم یک نمونه تصادفی را انتخاب می‌کند. خطوط 6.b تا 6.d، تعداد B تا از نزدیک‌ترین همسایگان هم‌رده با نمونهٔ انتخاب‌شده به نام Hits و B تا از نزدیک‌ترین همسایگان رده‌های دیگر را که Misses نامیده می‌شود، می‌یابند. در خط 6.e یک تابع ارزیابی (δ) وزن ویژگی‌ها را محاسبه می‌کند. این تابع تفاوت بین نمونه انتخاب‌شده و مجموعه‌های Hits و Misses را محاسبه می‌کند. ReliefF اصلی از فاصله منهتن در تابع (δ) استفاده می‌کند. معادله $pc/(1-p(class(x_t)))$ در فرمول وزن ویژگی، برای استفاده از ReliefF در مسائل چند رده استفاده می‌شود [25] . درنهایت، الگوریتم فهرستی از وزن ویژگی‌ها را به‌دست می‌آورد. بر اساس این فهرست، ویژگی‌های با وزن بیشتری انتخاب شده و مابقی حذف می‌شوند [24] .

Algorithm ReliefF (input:X, L, B)

1.*/ مجموعه داده آموزشی **=X / ***

1./*X=the set of training examples */

2.*/ تعداد نمونه‌هایی که باید به‌طور تصادفی انتخاب شوند**=L */**

2./*L=the number of random examples to draw */

3.*/ تعداد نزدیکترین همسایه‌هایی که باید محاسبه شوند**=B */**

3. /*B=the number of nearest neighbours to compute*/

4. برای هر ویژگی **k: {w_k=0.0}**

4. For each feature k{w_k=0.0}

۵.برای هر کلاس **c**{کسری از مجموعه‌داده که متعلق به این رده است **=:pc}**

5. For each class c{pc:=the fraction of X belonging to class c}

6. حلقه تکرار از ۱ تا **L**

6. For l:=1 to L do

a. یک نمونه تصادفی انتخاب کن**(x_t,y_t)**

a) Randomly select an instance (x_t,y_t)

b. مجموعه **hits** که مجموعه‌ای از نزدیکترین نمونه‌ها **(x_i,y_i)** به نمونه x_t است را بیاب به گونه‌ای که $y_i=y_t$

b) Let hits be the set of B examples (x_i,y_i) nearest to x_t such that $y_i=y_t$

c. برای هر رده $c\neq y_t$

c) For each class $c\neq y_t$

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

مجموعه **misses** که مجموعه‌ای از نزدیکترین نمونه‌ها **(x_i,y_i)** به نمونه x_t است را بیاب به گونه‌ای که y_j=**c**

Let misses be the set of B examples (xj,yj) Nearest to xt such that yj=c

d. پایان حلقه

d) End for

e. برای هر ویژگی **k**

e) For each feature k

$$w_k = w_k - \frac{1}{LB}\sum_{(xi,yi)\in Hit}\delta(x_{t,k},x_{i,k}) + \sum_{c\neq y_t}\frac{pc}{(1-p(class(x_t)))LB}\sum_{(xi,yi)}\delta(x_{t,k},x_{j,k})$$

f. پایان حلقه

f) End for

7. پایان حلقه

7. End for

۸. مجموعه مقادیر وزن‌ها **(W)** را به‌عنوان خروجی برگردان.

8. Return the set of weights (W)

9. پایان **ReliefF**

9. End Relief

(شکل-۱): شبه کد الگوریتم ReliefF

(Figure-1): Pseudo code of ReliefF

۲-۲- انتخاب هم‌زمان ویژگی‌ها و نمونه‌ها

بعضی از الگوریتم‌ها به جای انتخاب، فقط ویژگی‌ها یا نمونه‌ها می‌توانند هم ویژگی‌ها و هم نمونه‌ها را انتخاب کنند. این انتخاب می‌تواند در یک یا چند مرحله انجام شود. اگر ویژگی‌ها و نمونه‌ها به‌طور هم‌زمان انتخاب شوند، انتخاب در یک مرحله است؛ در غیر این صورت، در چند مرحله خواهد بود. به‌عنوان مثال، در [1] یک الگوریتم تکاملی برای انتخاب ویژگی و نمونه مورد استفاده قرار می‌گیرد. در این الگوریتم، هر کروموزوم دارای n + m ژن است (اگر مجموعه‌داده دارای n نمونه و m ویژگی باشد). اگر نمونه / ویژگی در مجموعه‌داده‌هایی که کروموزوم آن را نشان می‌دهد باقی بماند، مقدار ژن برابر با یک است، و اگر از مجموعه‌داده حذف شود، مقدار ژن برابر با صفر خواهد بود. برای هر ترکیب ممکن از نمونه‌ها و ویژگی‌ها، یک کروموزوم وجود دارد. یک رده‌بند برای ارزیابی کروموزوم‌ها استفاده می‌شود. اگر چه الگوریتم بهترین نمونه‌ها و ویژگی‌ها را در یک مرحله و با دقت زیادی انتخاب می‌کند، به‌دلیل استفاده از یک رده‌بند برای ارزیابی هر کروموزوم، برای مجموعه‌داده‌های بزرگ، این روش بسیار زمان‌بر یا غیر قابل استفاده خواهد بود. این مقاله بر روی هجده مجموعه‌داده اجرا شده که اگرچه ادعا شده شش عدد از آنها در مقیاس بزرگ هستند؛ اما بزرگترین آن‌ها (Satimage) شامل ۶۴۳۵ نمونه و ۳۶ ویژگی است که از مجموعه‌داده‌های استفاده‌شده در این مقاله بسیار کوچک‌تر است.

برخی از الگوریتم‌ها ویژگی‌ها را در چند مرحله انتخاب می‌کنند. در [26] یک راه‌کار تئوری مبتنی بر اطلاعات برای انتخاب نمونه‌ها و ویژگی‌ها و استفاده از آن‌ها در یک سامانه فیلترینگ تطبیقی[1] معرفی شده است. دلیل استفاده از انتخاب نمونه در این مقاله این است که تعداد مشتریان (نمونه‌ها) در سامانه پالایش گروهی افزایش می‌یابد؛ بنابراین، کاهش تعداد مشتریان، دقت و مقیاس‌پذیری سامانه را افزایش می‌دهد. همچنین با توجه به نوع داده‌های خاصی مسأله، از آنجایی که شناسایی ویژگی‌های مرتبط از ویژگی‌های نامناسب دشوار است، آن‌ها به جای انتخاب ویژگی، از وزن‌گذاری ویژگی‌ها استفاده می‌کنند؛ بنابراین، اطلاعات متقابل برای وزن‌دهی ویژگی‌ها استفاده می‌شود. برای انتخاب نمونه، ضریب همبستگی هر نمونه به‌کمک سایر نمونه‌ها محاسبه می‌شود. این مقاله به کاهش داده‌ها در یک حوزه خاص می‌پردازد و روش استفاده‌شده قابل تعمیم نیست. نتایج مقاله نیز فقط بر روی مجموعه‌داده EachMovie آزمایش شده است. هم‌چنین، اگرچه این روش دقت را افزایش می‌دهد، زمان لازم برای اجرای الگوریتم بسیار بیشتر از سایر روش‌ها است.

در [27] یک مدل کاهش جدید برای بهبود کارایی سامانه‌های تشخیص نفوذ پیشنهاد شده که در آن از OneR برای کاهش ویژگی و از انتشار وابستگی[2] برای انتخاب نمونه استفاده شده است. آنها از چارچوب برنامه‌نویسی موازی MapReduce [28] برای پیاده‌سازی الگوریتم خود برای افزایش سرعت کاهش داده‌ها استفاده کردند. برای آزمایش نتایج کاهش داده‌ها، داده‌های کاهش‌یافته به‌وسیلهٔ الگوریتم‌های KNN و SVM رده‌بندی شدند. این مقاله بر

[1] Collaborative filtering system
[2] Affinity propagation

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]
[DOI: 10.29252/jsdp.17.4.49]

روی دو مجموعه‌دادهٔ KDD99 و CDMC2012 آزمایش و اجرا شد. نویسندگان ادعا کردند که سرعت تشخیص حمله ۱۸۴ بار افزایش یافته است، درحالی‌که دقت تشخیص تفاوت معنی‌داری ندارد.

یک روش برای ایجاد یک سامانه تشخیص خودکار مبتنی بر تصویر در [29] معرفی شد. ایجاد یک سامانه تفسیر خودکار برای تشخیص بیماری‌ها نیازمند یک کاتالوگ تصویری است که به‌عنوان پایه‌ای برای توسعه سامانه استفاده می‌شود؛ بنابراین، نویسندگان این مقاله با استفاده از روش‌های انتخاب ویژگی و انتخاب نمونه، نمونه‌ای از تصاویر را ایجاد کردند. آنها روش خود را بر روی چهار کاتالوگ تصویر اجرا کردند. برای کاهش نمونه‌ها، الگوریتم پیشنهادی هر نمونه را با همسایه‌های هم‌کلاسش ترکیب و یک نمونه جدید ایجاد می‌کند. عمل ادغام نمونه‌ها تا زمانی ادامه می‌یابد که نمونه‌های جدید دقت رده‌بندی را کاهش ندهند. برای کاهش ویژگی‌ها، الگوریتم از یک روش بسته‌بندی برای تعیین وزن و انتخاب ویژگی‌ها استفاده می‌کند. در این بخش، هر ویژگی با رده‌بند KNN ارزیابی و ویژگی‌های خوب تعیین می‌شود. درواقع، در هر دو روش انتخاب نمونه و ویژگی، الگوریتم‌های رده‌بندی به‌عنوان تابع ارزیابی استفاده می‌شود. شاید این روش دقت سامانه را افزایش دهد، اما زمان زیادی طول می‌کشد و برای مجموعه‌داده‌های حجیم قابل استفاده نیست.

در [30] یک الگوریتم ژنتیک برای انتخاب ویژگی‌ها و نمونه‌ها مورد استفاده قرار می‌گیرد. در این مقاله، انتخاب ویژگی و نمونه به‌ترتیب، نه هم‌زمان انجام می‌شود. نویسندگان از روش‌های کاهش داده موجود بر پایه الگوریتم ژنتیک برای کاهش ویژگی‌ها و نمونه‌ها استفاده کرده‌اند. بسته به اینکه آیا ویژگی‌ها یا نمونه‌ها و با چه ترتیبی انتخاب شوند، پنج روش مختلف به مجموعه‌داده‌ها اعمال شده است. این روش‌ها دو استراتژی را بر اساس نتایج به‌دست‌آمده از رده‌بندی داده‌های کاهش‌یافته با دو رده‌بند KNN و SVM پیشنهاد کردند. نتیجه کلی این است که انجام انتخاب ویژگی و سپس انتخاب نمونه‌ها، راه حل بهینه برای پیش‌پردازش داده‌ها است؛ اما برای پایگاه داده‌های بزرگ، این همیشه بهترین حالت نیست، و گاهی اوقات انجام انتخاب نمونه و سپس انتخاب ویژگی بهتر است. البته این نتیجه‌گیری بر اساس نتایج به‌دست‌آمده بر روی هشت مجموعه داده، به‌دست آمده است؛ اما نمی‌توان آن را قطعی دانست و به‌عنوان یک قانون کلی در مورد تمام مجموعه داده‌ها اعمال کرد.

یک روش انتخاب ویژگی و نمونه به‌طور هم‌زمان برای ارائه یک صفحه داده فشرده و دقیق برای طبقه‌بندی متن در [31] استفاده می‌شود. الگوریتم پیشنهادشده در دو مرحله عمل می‌کند: در مرحلهٔ نخست، ویژگی‌هایی که به‌طور دقیق رده هدف را پیش‌بینی می‌کنند، به‌ترتیب انتخاب، سپس تمام اسنادی که دست‌کم یکی از این ویژگی‌ها را ندارند از مجموعه آموزش حذف می‌شوند. در مرحله دوم، تعدادی از ویژگی‌ها از میان ویژگی‌های انتخاب‌شده در مرحله قبلی، انتخاب می‌شوند. آنها تمایل دارند مکمل رده هدف را پیش‌بینی کنند. مجموع اسناد و ویژگی‌های انتخاب‌شده در این دو مرحله، مجموعه آموزشی را ایجاد می‌کند. نویسندگان از مجموعه‌ای از داده‌های متنی که شامل بیست‌هزار پست خبری بود، برای پیاده‌سازی و مقایسه روش پیشنهادشده با روش‌های دیگر استفاده می‌کنند.

در [32] پژوهش‌گران از کاهش هم‌زمان داده‌ها برای طراحی یک مدل پیش‌بینی ورشکستگی در موضوعات اقتصادی استفاده کردند. آنها از استدلال مبتنی بر مورد (CBR) برای پیش بینی ورشکستگی استفاده کردند. با وجود تمام محاسن، کارایی آن بسیار کم است. آنها از انتخاب ویژگی‌ها و موارد بر اساس الگوریتم ژنتیک برای بهبود عملکرد CBR استفاده کردند. این روش فقط برای یک برنامه خاص طراحی شده است. همچنین، به‌دلیل استفاده از الگوریتم ژنتیک، این روش بسیار وقت‌گیر است و نیاز به منابع سخت‌افزاری خاص دارد. همچنین اندازه جمعیت و تعداد نسل‌هایی که برای الگوریتم ژنتیک تعریف می‌شود، باید کوچک باشد؛ در غیر این صورت فضای جستجو بسیار بزرگ خواهد بود. مجموعه‌داده آن‌ها شامل اطلاعات وضعیت اقتصادی مربوط به ۲۶۷۰ شرکت است.

به‌طور خلاصه، کاهش هم‌زمان ویژگی‌ها و نمونه‌ها می‌تواند یک مسأله مهم باشد و به افزایش کارایی و صحت رده‌بندها کمک کند. همان‌طور که در این بخش مشاهده شد، پژوهش‌های زیادی در این زمینه صورت گرفته است، اما متأسفانه راه‌حل‌های پیشنهادی دچار برخی مشکلات هستند. این مشکلات شامل زمان‌بربودن، قابل استفاده‌نبودن در مجموعه داده‌های بزرگ، مناسب‌بودن برای برخی از مشکلات خاص و تعمیم ناپذیری است؛ بنابراین، این مقاله تلاش می‌کند تا با برطرف‌کردن این مشکلات، با ارائه یک الگوریتم مناسب برای داده‌های بزرگ و از انواع مختلف تلاش کند. همچنین از ساختارهای برنامه‌نویسی موازی برای

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

کاهش زمان اجرای الگوریتم پیشنهادی استفاده و در بخش بعدی، الگوریتم پیشنهادی توضیح داده شده است.

۳- الگوریتم پیشنهادی (IFSB-ReliefF)

این بخش نحوه استفاده از الگوریتم ReliefF برای انتخاب هم‌زمان ویژگی‌ها و نمونه‌ها را توضیح می‌دهد. همان‌طورکه درقبل گفته شد، الگوریتم ReliefF یک الگوریتم رتبه‌بندی ویژگی است که بر اساس تابع ارزیابی برای هر یک از ویژگی‌ها وزنی اختصاص می‌دهد. در این الگوریتم، یک نمونه تصادفی در هر تکرار انتخاب و تفاوت آن با مجموعه Hit و Miss اندازه‌گیری می‌شود. یک ویژگی خوب، ویژگی‌ای است که بیشتر شبیه به نزدیک‌ترین نمونه از همان رده است و بیشترین تفاوت را با نزدیک‌ترین نمونه از رده‌های مخالف دارد. الگوریتم IFSB-ReliefF از همین روش برای انتخاب نمونه‌ها و ویژگی‌ها استفاده می‌کند. شکل (۲) شبه‌کد IFSB-ReliefF را نشان می‌دهد.

IFSB-ReliefF Algorithm (input:X, B)

1. /*X=the set of training examples */
۱. */ مجموعه داده آموزشی =X */
2. /*B=the number of nearest neighbors to compute*/
۲. */ تعداد نزدیک‌ترین همسایه‌هایی که باید محاسبه شوند=B */
3. /*L= Number of instances.*/
۳. */ تعداد نمونه‌ها =L*/
4. For t:=1 to L do (This loop can run in parallel)
۴. حلقه تکرار برای t=1 تا L (این حلقه می‌تواند به طور موازی اجرا شود)
a) Find B nearest instances to x_t from class y_t (Hit set) by Jaccard index or Manhattan distance
a. تعداد B تا از نزدیکترین نمونه‌ها به x_t از رده y_t (مجموعه Hit) را با شاخص ژاکارد یا فاصله منهتن بیاب.
b) For each class $c \neq y_t$
b. برای هر رده $c \neq y_t$
Find B nearest instances to xt from class c and add to Miss set by Jaccard index or Manhattan distance
تعداد B تا از نزدیک‌ترین نمونه‌ها به x_t از هر رده c را با شاخص ژاکارد یا فاصله منهتن بیاب و به مجموعه Miss اضافه کن.
c) End for (line 4.b)
c. پایان حلقه (خط 4.b)
d) Calculate the weight of each instance x_t and put it in array W:
d. وزن هر نمونه x_t را محاسبه کن و آن را در آرایه W قرار بده.

$$\mathbf{w_t} = \sum_{\mathbf{c \neq y_t}} \frac{\mathbf{1}}{\mathbf{LB}} \sum_{\mathbf{(xj,yj) \in Miss}} \boldsymbol{\delta}(\mathbf{x_t}, \mathbf{x_j}) - \frac{\mathbf{1}}{\mathbf{LB}} \sum_{\mathbf{(xi,yi) \in Hit}} \boldsymbol{\delta}(\mathbf{x_t}, \mathbf{x_i})$$

e) For each feature k of x_t calculate the weight and put it in array WF:
e. برای هر ویژگی k از نمونه x_t وزن را محاسبه کن و آن را در آرایه WFI قرار بده.

$$\boldsymbol{wf_{tk}} = \sum_{\boldsymbol{c \neq y_t}} \frac{\boldsymbol{pc}}{(\mathbf{1} - \boldsymbol{p}(\boldsymbol{class}(\boldsymbol{x_t})))\boldsymbol{LB}} \sum_{\mathbf{(xj,yj) \in Miss}} \boldsymbol{\delta}(\boldsymbol{x_{t,k}}, \boldsymbol{x_{j,k}}) - \frac{\mathbf{1}}{\boldsymbol{LB}} \sum_{\boldsymbol{(xi,yt) \in Hit}} \boldsymbol{\delta}(\boldsymbol{x_{t,k}}, \boldsymbol{x_{i,k}})$$

f) End for (line 4.e)
f. پایان حلقه (خط 4.e)
5. End for (line 4)
۵. پایان حلقه (خط 4)
6. Calculate the average of all Wf_{tk} and get a weight for each feature Wf.
۶. به‌ازای هر ویژگی k، میانگین وزن تمام مقادیر موجود در WFI را محاسبه کن و یک وزن برای هر ویژگی به‌دست آور و در WF قرار بده.
7. Sort the weight array W and Wf.
۷. آرایه وزن W و WF را مرتب کن.
8. Based on the sorted array W, select instances that have more weight from each class according to the proportion to be selected and put in the output array.
۸. بر اساس آرایه مرتب شده W، از هر کلاس با توجه به pc (احتمال پیشین) آن نمونه‌هایی که وزن بیشتری دارند را انتخاب کرده و در آرایه Out قرار بده.
9. In the output array, based on WF array, keep the features with more weight and remove the rest.
۹. بر اساس آرایه WF، ویژگی‌های با وزن بیشتر را از آرایه Out انتخاب کرده و در آرایه خروجی قرار بده.
10 . پایان الگوریتم پیشنهادی
۱۰. End Proposed Algorithm.

(شکل-۲): الگوریتم IFSB-ReliefF

(Figure-2): Pseudo code of IFSB-ReliefF

همان‌طورکه در شکل نشان داده شده، IFSB-ReliefF به‌شرح زیر عمل می کند: در خط ۴، یک حلقه تکرار به تعداد همه نمونه‌ها وجود دارد، اما در الگوریتم ReliefF، تنها چند نمونه تصادفی

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]
[DOI: 10.29252/jsdp.17.4.49]

انتخاب می‌شود. اگر چه این کار وقت‌گیر خواهد بود، چون دستورهای داخل حلقه به‌طور مستقل برای یک نمونه اجرا می‌شوند، می‌توانند به‌طور موازی در سیستم‌های چندپردازنده اجرا شوند. چنانچه در بخش نتایج، ملاحظه خواهد شد، این امر به کاهش زمان اجرا و امکان اجرای الگوریتم روی مجموعه‌داده‌های بزرگ، کمک زیادی خواهد کرد. مانند ReliefF در خطوط 4.a و 4.b، تعدادB تا برای مجموعه Hits وB تا برای مجموعه Missses از هر رده دیگر انتخاب شده است. IFSB-ReliefF می‌تواند از هر فرمول فاصله برای تابع ارزیابی (δ) برای اندازه‌گیری فاصله بین نمونه‌ها استفاده کند. همان‌گونه که پیش‌تر گفته شد ReliefF اولیه [21] از تابع فاصله منهتن برای اندازه‌گیری فاصله بین دو نمونه استفاده می‌کند، اما از آنجا که الگوریتم ReliefF اندازه فاصله بین دو نمونه را برای به‌دست‌آوردن ارزش و وزن ویژگی‌ها استفاده می‌کند و بنابراین، به این‌که به‌طور کلی دو نمونه چقدر با هم تشابه دارند، کاری ندارد؛ اما در این مقاله، از تابع فاصله منهتن و شاخص ژاکارد به‌عنوان تابع ارزیابی استفاده می‌شود. گفتنی است که برای نخستین‌بار در مقاله [33]، توسط نویسندگان همین مقاله از شاخص ژاکارد به‌عنوان تابع ارزیابی در ReliefF استفاده می‌شود. شاخص ژاکارد، که همچنین به‌عنوان ضریب تشابه ژاکارد شناخته می‌شود، برای مقایسه شباهت دو مجموعه استفاده می‌شود؛ بنابراین می‌توان مقایسه کرد که آیا دو نمونه به‌طور کلی با هم تشابهی دارند یا خیر. اگر x=(x1,x2,...,xn) و y=(y1,y2,...,yn) دو بردار با اعداد حقیقی بزرگ‌تر از صفر باشند، شاخص ژاکارد از آنها به‌صورت معادله (۱) تعریف می‌شود:

$$\mathbf{J(X,Y)} = \frac{\sum_i \mathbf{min(x_i,y_i)}}{\sum_i \mathbf{max(x_i,y_i)}} \qquad (۱)$$

از آنجا که این فرمول برای اعداد حقیقی بیشتر از صفر تعریف می‌شود. برای استفاده از این فرمول در تابع، ویژگی‌های عددی برخی از مجموعه‌های داده ابتدا باید به فاصله‌ای از صفر تا یک نرمال‌سازی شوند؛ بنابراین نتایج معادله (۱) نیز در محدوده صفر تا یک قرار می‌گیرند. اگر نتیجه یک باشد، دو نمونه به‌طوردقیق همانند هستند و اگر نتیجه صفر باشد، به این معنی است که آنها به‌طورکامل متفاوت هستند و بین آنها هیچ شباهتی وجود ندارد.

برای دو مجموعه غیر عددی (رشته‌ها) x=(x1,x2,...,xn) و y=(y1,y2,...,yn)، شاخص ژاکارد به‌صورت معادله (۲)تعریف می‌شود:

$$\mathbf{J(X,Y)} = \frac{|\mathbf{X \cap Y}|}{|\mathbf{X \cup Y}|} \qquad (۲)$$

برخی از ویژگی‌های مجموعه‌داده‌ها مقادیر مفقود دارند؛ در این حالت، اگر مقدار xi یا yi یا هر دو ناشناخته باشد، آنگاه $\mathbf{x_i \cap y_i} = \emptyset$. IFSB-ReliefF از معادله (۱) و معادله (۲) به‌عنوان یک تابع ارزیابی استفاده می‌کند. اگر ویژگی‌ها عددی باشند، معادله (۱) و اگر غیر عددی یا دارای مقادیر مفقود باشند، معادله (۲) استفاده می‌شود.

برای مثال اگر سه نمونه با ویژگی‌های زیر در مجموعه‌داده موجود باشد:

نمونه ۱ (I1) : 0.9 , 0 , ? , c

نمونه ۲ (I2): 0.15 , 0.5 , a , b

نمونه ۳ (I3) : 0.1 , 0.5 , a , ?

برای محاسبه شاخص ژاکارد برای نمونه ۱و ۲ و نمونه ۲ و ۳ به این شکل عمل می‌شود که ابتدا ویژگی‌های عددی و غیرعددی جدا و برای هر کدام با فرمول مخصوص محاسبه صورت گرفته و در اخر بر حسب تعداد ویژگی‌های عددی و غیرعددی، میانگین گرفته می‌شود.

$$\mathrm{J(I1,I2)_{num}} = \frac{\min(0.9,0.15)+\min(0,0.5)}{\max(0.9,0.15)+\max(0,0.5)} = \frac{0.15+0}{0.9+0.5} = 0.107$$

$$\mathrm{J(I1,I2)_{Str}} = \frac{|\emptyset|}{|(a,b,c,?)|} = 0$$

$$\mathrm{J(I1,I2)} = \frac{0.107+0}{2} = 0.053$$

$$\mathrm{J(I2,I3)_{num}} = \frac{\min(0.15,0.1)+\min(0.5,0.5)}{\max(0.15,0.1)+\max(0.5,0.5)} = \frac{0.1+0.5}{0.15+0.5} = 0.92$$

$$\mathrm{J(I2,I3)_{Str}} = \frac{|(a)|}{|(a,b,?)|} = 0.333$$

$$\mathrm{J(I2,I3)} = \frac{0.92+0.333}{2} = 0.628$$

با توجه به مثال، مشاهده می‌شود که از نظر شاخص ژاکارد دو نمونه ۲ و ۳ تشابه بیشتری به هم نسبت به نمونه‌های ۱و۲ دارند.

برای هر نمونه، الگوریتم نمونه‌هایی را برای مجموعه Hitو Miss پیدا می کند که دارای بیشینه مقدار شاخص ژاکارد هستند. درنهایت، در خط 4.d، وزن هر نمونه با استفاده از مقادیر مجموعهHit وMiss محاسبه می‌شود. همان‌طور که مشاهده می‌شود، هر قدر یک نمونه مشابه نمونه‌هایی از سایر رده‌ها است، وزن آن افزایش می‌یابد. در مقابل، اگر نمونه شبیه به نمونه‌های رده خودش باشد، وزن کاهش می‌یابد. دلیل این افزایش و کاهش این است که در الگوریتم پیشنهادی، نمونه‌های بحرانی در نقاط مرزی بین دو

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

رده برای تشخیص مهم‌تر هستند. درحقیقت، نمونه‌هایی که بسیار شبیه یکدیگرند، نمونه‌های تکراری هستند. آنها فقط اندازه مجموعه‌داده را افزایش می‌دهند و عملکرد و دقت طبقه‌بندی را بهبود نمی‌دهند.

در خط 4.e، با استفاده از مقادیر مجموعهHit و Miss، وزن هر ویژگی نیز محاسبه می‌شود. گفتنی است که pc و $p(class(x_t))$ احتمال پیشین هر رده، یعنی نسبت نمونه‌های هر رده به کل نمونه‌ها، است. این احتمال باید قبل از شروع الگوریتم محاسبه شود. پس از پایان حلقه اصلی برنامه در خط ۵، دو آرایه از وزن وجود دارد، یکی برای نمونه (W) و یکی برای ویژگی‌های هر نمونه (WFI).

در خط ۶ برای به‌دست‌آوردن وزن نهایی هر ویژگی، میانگین مقادیر آرایه WFI برای هر ویژگی محاسبه می‌شود و در آرایه WF قرار می‌گیرد. در خط ۷، مقادیر آرایه‌های W و WF برای شناسایی نمونه‌ها و ویژگی‌های بهتر، مرتب می‌شوند.

در خط ۸، بر اساس pc، نمونه‌هایی با وزن بالاتر انتخاب می‌شوند. یکی از اصلاحاتی که در IFSB-ReliefF انجام شده، درنظرگرفتن نسبت نمونه‌های انتخاب‌شده از هر رده نسبت به همه نمونه‌های آن رده است. دلیل این امر این است که برخی از الگوریتم‌ها، مانند الگوریتم All k-NN [34]، احتمال پیشین هر رده را در نظر نگرفته و انتخاب کورکورانه‌ای انجام می‌دهند. این ممکن است باعث عدم تعادل بین نسبت رده‌ها شود. همچنین در برخی موارد، که در آن یک رده فقط چند نمونه دارد، ممکن است، تمام نمونه‌های آن حذف شده و هیچ ردی از این رده باقی نماند؛ بنابراین، این بهبود در IFSB-ReliefF به حفظ وضعیت اولیه مجموعه‌داده‌ها کمک می‌کند. البته، روش محاسبه را می‌توان برای نگهداری تعداد بیشتری از نمونه‌های رده اقلیت و حل مشکل مجموعه‌داده‌های نامتوازن استفاده کرد. این کار می‌تواند با محاسبه احتمال پیشین هر رده و با انتخاب نمونه‌هایی با وزن بیشتری از هر رده انجام شود. تعداد نمونه‌های انتخاب‌شده از هر رده بستگی به میزان کاهش دارد. درنهایت، در خط ۹، فیلترکردن ستون‌ها (ویژگی‌ها) در مجموعه نمونه انتخاب‌شده انجام می‌شود و ویژگی‌هایی که وزن بیشتری دارند، حفظ شده و بقیه حذف می‌شوند.

خروجی الگوریتم IFSB-ReliefF یک مجموعه‌داده است که هم از نظر تعداد سطرها و هم از نظر تعداد ستون‌ها کاهش یافته است. به‌منظور بررسی موفقیت الگوریتم کاهش داده، باید الگوریتم رده‌بندی بر روی داده‌های کاهش‌یافته اجرا شود؛ سپس نتایج رده‌بندی داده‌های کاهش‌یافته با IFSB-ReliefF با نتایج به‌دست‌آمده از سایر الگوریتم‌های کاهش مقایسه می‌شود. برای آزمایش نتایج الگوریتم، چندین مجموعه‌داده استفاده شده که خصوصیات و نتایج آن‌ها در بخش بعدی توضیح داده خواهد شد.

۴- نتایج تجربی

در این بخش، نتایج رده‌بندی داده‌های انتخاب‌شده به‌وسیلهٔ IFSB-ReliefFبا رده‌بندی داده‌های اولیه و برخی از الگوریتم‌های دیگر کاهش داده مقایسه می‌شود. ابتدا برخی از مسائل مانند نیازمندی‌های سخت‌افزاری و نرم‌افزاری، ویژگی‌های مجموعه‌داده‌ها، پارامترهای الگوریتم و معیارهای ارزیابی مورد استفاده برای ارزیابی توصیف، سپس با استفاده از الگوریتم رده‌بندی برای کاهش داده‌ها، عملکرد الگوریتم‌ها در چهار مجموعه داده ارزیابی می‌شود؛ درنهایت، نتایج به‌دست‌آمده ارائه و در مورد آن‌ها بحث خواهد شد. برای آزمایش IFSB-ReliefF، مجموعه‌داده‌های استفاده‌شده شامل انواع داده‌های مختلف (عددی و غیرعددی) و مقادیر مفقود است. مشخصات مجموعه‌داده‌های استفاده‌شده در جدول (۱) ارائه شده است.

(جدول-۱): مشخصات مجموعه‌های داده
(Table-1): 1 Datasets characteristics

مجموعه داده	تعداد نمونه	تعداد ویژگی	تعداد کلاس	نوع ویژگی	دارای مقادیر مفقود
MNIST (test part)[1]	50,000	784	10	عددی	خیر
Balance-Scale2	625	4	3	عددی	خیر
Census[3]	30,000	41	2	عددی و غیر عددی	بله
German credit4	1000	20	2	عددی و غیر عددی	خیر

مشخصات سخت‌افزاری و نرم‌افزاری سامانه عبارتند از:

- پردازنده: Intel® Xeon® CPU E5-2620 v4 @ 2.10GHz (2 processors)

[1]http://www.cs.ubc.ca/labs/beta/Projects/autoweka/datasets/mnist.zip
[2] http://archive.ics.uci.edu/ml/datasets/balance+scale
[3] https://archive.ics.uci.edu/ml/datasets/Census-Income+%28KDD%29
[4]https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]
[DOI: 10.29252/jsdp.17.4.49]

- RAM: ۶۴ گیگابایت
- سیستم عامل: Win 7 64 bit
- نرم‌افزارهای مورد استفاده برای کاهش داده‌ها با سایر الگوریتم‌های کاهش و هم‌چنین رده‌بندی داده‌ها: Rapid Miner و WEKA 3.8.
- نرم‌افزار مورد استفاده برای اجرای IFSB ReliefF: Matlab r2013.

برای ارزیابی نتایج IFSB-ReliefF، آن‌ها با نتایج برخی از الگوریتم‌های انتخاب نمونه و ویژگی مقایسه شدند. به همین ترتیب، نتایج رده‌بندی داده‌های کاهش‌یافته با نتایج مجموعه داده‌های اصلی مقایسه شد.

الگوریتم مورد استفاده برای انتخاب نمونه‌ها All k-NN است. در All k-NN، تمام نمونه‌هایی که به‌درستی به‌وسیلهٔ نزدیک‌ترین i تعداد همسایگانشان رده‌بندی نشده‌اند (برای i = 1 .. k) حذف می‌شوند. الگوریتم ReliefF اولیه نیز برای انتخاب ویژگی‌ها استفاده می‌شود؛ علاوه‌براین، چون IFSB-ReliefF هم‌زمان کاهش نمونه‌ها و ویژگی‌ها را انجام می‌دهد، ترکیبی از ReliefF و All k-NN نیز بر روی داده‌ها انجام می‌شود. یعنی، در ابتدا، تعدادی از نمونه‌ها با استفاده از All k-NN انتخاب شده و سپس از میان نمونه‌های انتخاب‌شده، تعدادی از ویژگی‌ها بر اساس میزان کاهش مورد نظر انتخاب و بقیه حذف می‌شوند.

جدول (۲) پارامترهای هر الگوریتم را نشان می‌دهد. گفتنی است که نرخ کاهش ۵۰٪ و مقدار L به‌صورت تجربی انتخاب شده و سایر پارامترها پیش‌فرض‌های مورد استفاده در WEKA و Rapid Miner است. البته این مقادیر پیش‌فرض بر اساس مقالات متعدد و مقدار استفاده‌شده در بیش‌تر آنها است. برای مثال در [24] گفته شده است که برای بیش‌تر اهداف انتخاب مقدار B=10 مطمئن خواهد بود.

(جدول-۲): .پارامترهای الگوریتم‌ها- The algorithms parameters
(Table-2): Algorithm parameters.

الگوریتم‌ها	پارامترها
All k-NN	**k start:3 k stop:10**
ReliefF	تعداد نمونه‌های انتخاب شده **(L): 50%** نمونه‌ها تعداد همسایه‌ها **(B): 10** تعداد ویژگی‌های انتخاب شده بعد از رتبه بندی ویژگی‌ها: **50%** ویژگی‌ها

برای ارزیابی کیفیت نتایج حاصل از IFSB-ReliefF، مجموعه‌داده‌های کاهش‌یافته به‌وسیلهٔ الگوریتم C4.5 رده‌بندی می‌شود. این الگوریتم یک درخت تصمیم را تولید می‌کند که می‌تواند مجموعه داده‌هایی با ویژگی‌های گسسته و پیوسته و مقادیر مفقود را مدیریت کند [35]. C4.5 از آنتروپی اطلاعات برای ایجاد درخت استفاده می‌کند. C4.5، در هر گره، یک ویژگی را انتخاب می‌کند که نمونه‌های هر رده را به‌طور مؤثرتر جدا می‌کند. در تمام آزمایش‌ها (به جز دو آزمایش مقایسه با مقالات دیگر در جدول (۵) و جدول (۸))، ۶۶٪ از داده‌ها برای یادگیری و بقیه برای آزمایش استفاده شدند. این یکی از روش‌های استاندارد برای آموزش و آزمایش سامانه‌ها در WEKA است. البته روش‌های دیگری مانند K-fold cross validation نیز وجود دارند که برای این‌کار استفاده می‌شوند، اما به‌دلیل حجم زیاد داده‌ها، به‌ویژه در مجموعه‌داده‌های اولیه و بدون کاهش، استفاده از آن‌ها موجب افزایش زمان رده‌بندی یا عدم امکان اجرا و بروز خطا می‌شود. فقط در مورد مقایسه دو مجموعه داده german credit و Balance-scale (در جدول (۵) و جدول (۸)) از روش 10-fold cross validation استفاده شده است.

برای ارزیابی نتایج، معیارهای کارایی زیر به‌کار رفته‌اند:

- نرخ کاهش: این معیار نرخ کاهش‌داده را مشخص کرده و مشخص می‌کند چه مقدار از مجموعه‌داده حذف شده است. و با فرمول زیر به‌دست می‌آید [36, 1]:

نرخ کاهش= اندازه مجموعه کاهش‌یافته/ اندازه مجموعه اولیه. (۳)

در این مقاله این معیار با واحد درصد نشان داده می‌شود. گفتنی است که در بسیاری از الگوریتم‌های کاهش داده مانند All k-NN، نمی‌توان نرخ کاهش داده را از ابتدا مشخص کرد و خود الگوریتم با توجه به پارامترهایی تصمیم می‌گیرد که چه داده‌ای را حذف کند. هم‌چنین، چون الگوریتم‌های ReliefF و All k-NN فقط کاهش ویژگی یا کاهش نمونه را انجام می‌دهند، حتی اگر نرخ کاهش آن‌ها با IFSB-ReliefF برابر باشد، نمی‌تواند نشان‌دهنده کارایی یکسان آن‌ها باشد؛ بنابراین، نرخ کاهش الگوریتم می‌تواند به‌عنوان یک پارامتر مهم در انتخاب آن باشد. در عین حفظ کیفیت داده، هر قدر نرخ کاهش بیشتر باشد، نشان‌دهنده برتری الگوریتم خواهد بود.

- دقت: این معیار نشان‌دهندهٔ تعداد نمونه‌هایی که به‌طور صحیح رده‌بندی شده‌اند نسبت به کل نمونه‌ها است.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]
[DOI: 10.29252/jsdp.17.4.49]

- زمان اجرا: زمان لازم برای اجرای الگوریتم‌های کاهش را نشان می‌دهد. زمان با واحد ثانیه نشان داده می‌شود.
- شاخص Kappa: این شاخص برای اندازه‌گیری میزان تصادفی بودن نتایج است. مقدار آن بین ۱- تا ۱ متغیر است که مقدار صفر نشان‌دهنده تصادفی‌بودن نتایج است و هرچه به مقدار یک نزدیک باشد، احتمال تصادفی‌بودن نتایج کمتر است [1].

$$\text{Kappa} = \frac{n\sum_{i=1}^{c} x_{ii} - \sum_{i=1}^{c} x_{i.}x_{.i}}{n^2 - \sum_{i=1}^{c} x_{i.}x_{.i}} \qquad (۴)$$

در فرمول (۴)، x_{ii} تعداد سلول‌ها در قطر اصلی ماتریس اغتشاش، n تعداد نمونه‌ها، c تعداد رده‌ها و $x_{i.}$ و $x_{.i}$ ستون‌ها و سطرهای تعداد کل هستند.

- انحراف معیار (RMSE): RMSE تفاوت بین مقادیر پیش‌بینی‌شده با رده‌بند برای داده‌های کاهش‌یافته و نتایج واقعی را نشان می‌دهد [37]. درواقع، RMSE خطاهای پیش‌بینی رده‌بند را نشان می‌دهد. RMSE یک تخمین‌گر $\widehat{\boldsymbol{\theta}}$ نسبت به پارامتر تخمین زده‌شده $\boldsymbol{\theta}$ به‌صورت زیر تعریف می‌شود:

$$\text{RMSE}(\hat{\theta}) = \sqrt{\text{MSE}(\hat{\theta})} = \sqrt{\text{E}\left((\hat{\theta} - \theta)^2\right)}. \qquad (۵)$$

جدول (۳) نتایج مجموعه‌داده MNIST را نشان می‌دهد. همان‌گونه که در جدول دیده می‌شود، الگوریتم ReliefF تنها ویژگی‌ها را کاهش می‌دهد و دقت رده‌بندی تفاوت زیادی با مجموعه داده اولیه ندارد. هم‌چنین، All k-NN فقط نمونه‌ها را کاهش می‌دهد و نرخ کاهش آن هم بسیار پایین است و درواقع می‌توان گفت که همه نمونه‌ها حفظ شده‌اند. البته دقت رده‌بندی این الگوریتم بالاتر از مجموعه‌داده اولیه است؛ اما هدف کاهش داده برآورده نمی‌شود. همچنین اگر چه ترکیب ReliefF و All k-NN دقت را به میزان زیادی افزایش داده و خطای RMSE را کاهش می‌دهد، اما نرخ کاهش هم‌چنان رضایت‌بخش نیست. علاوه‌براین، زمان اجرای هر سه الگوریتم یادشده، به‌ویژه ترکیب ReliefF و All k-NN بسیار بیشتر از الگوریتم IFSB-ReliefF است. دلیل کاهش زمان IFSB-ReliefF، و کمتربودن آن حتی از ReliefF اولیه، استفاده از ساختار برنامه‌نویسی موازی است که به الگوریتم اجازه می‌دهد حلقه اصلی برنامه را به‌طور مستقل برای هر یک از نمونه‌ها اجرا کند. برای مثال، زمان اجرا برای IFSB-ReliefF با شاخص ژاکارد با نرخ کاهش ۷۵٪ روی مجموعه داده MNIST به‌صورت موازی 8671 ثانیه است، در صورتی که اگر الگوریتم به‌صورت غیرموازی اجرا شود، زمان اجرا ۵۴۹۲۳ ثانیه خواهد بود. همچنین در مورد تابع ارزیابی مورد استفاده در IFSB-ReliefF، نتایج شاخص ژاکارد و فاصله منهتن در نرخ کاهش ۷۵٪، از نظر میزان دقت رده‌بندی و RMSE به‌طورتقریبی مشابه هم هستند و فقط در نرخ کاهش ۹۱٪، فاصله منهتن عملکرد بهتری داشته است. از نظر میزان زمان مصرف‌شده نیز به‌طورتقریبی مشابه‌اند، و فقط روش ژاکارد، کمی زمان بیشتری صرف می‌کند.

در جدول (۴) نتایج مربوط به مجموعه‌داده German credit ارائه شده است. در این جدول نیز بهترین دقت مربوط به روش All k-NN است، اما همان‌گونه که مشاهده می‌شود، اگرچه زمان کمی برای اجرا صرف شده، نرخ کاهش تنها 30.2% است، در صورتی که روش پیشنهادی این مقاله با فاصله منهتن و نرخ کاهش 89.5% به نتایجی با شاخص kappa بالاتر و دقتی بسیار نزدیک به All k-NN دست یافته است. در این مجموعه داده، شاخص ژاکارد اگرچه در شاخص‌های رده‌بندی عملکرد ضعیف‌تری از فاصله منتهن داشته، اما زمان مورد نیاز برای کاهش آن، به‌طورتقریبی ۲/۵ برابر کمتر بوده و هم‌چنین، نتایج به‌دست‌آمده به‌وسیلهٔ روش IFSB-ReliefF با نتایج مقالات [1] و [30] در جدول (۵) مقایسه شده است. برای مقایسه بین روش‌ها از رده‌بند KNN با K=1استفاده شده است. البته مقاله [1]، مقدار K را به‌طور صریح بیان نکرده و ممکن است از مقادیر بیشتری برای K استفاده کرده باشد که تأثیر زیادی بر دقت به‌دست‌آمده دارد. برای مثال در روش IFSB-ReliefF اگر مقدار K=2 باشد، دقت به‌دست‌آمده برای نرخ تابع ارزیابی فاصله منهتن (مورد آخر جدول) ۷۱.۶۶ خواهد بود. متأسفانه هیچ‌کدام از این دو مقاله سخت‌افزار مورد استفاده خود را بیان نکردند تا زمان اجرای الگوریتم با توجه به آن سنجیده شود؛ اما با توجه به این‌که هر دو الگوریتم [1] و [30] از الگوریتم ژنتیک و رده‌بندها برای بررسی و کاهش نمونه‌ها و ویژگی‌ها استفاده می‌کنند، و الگوریتم‌های ژنتیک ذاتاً زمان زیادی را صرف ارزیابی یک جواب و محاسبه میزان بهینگی آن می‌کنند، زمان مورد استفاده آن‌ها به‌احتمال بیشتر از روش پیشنهادی خواهد بود. همچنین همان‌گونه که در جدول‌های (۴ و ۵) مشخص است، مقدار شاخص Kappa برای IFSB-ReliefF بیشتر از سایر روش‌ها است. این شاخص نشان‌دهنده تصادفی‌بودن یا نبودن نتایج است و هر چقدر به یک نزدیک باشد، یعنی نتایج آزمایش از ثبات بیشتری برخوردار است.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]
[DOI: 10.29252/jsdp.17.4.49]

(جدول-۳): نتایج اعمال الگوریتم‌های کاهش داده بر مجموعه‌داده MNIST

(Table-3): The results of applying data reduction algorithms on MNIST data set.

الگوریتم کاهش داده	نرخ کاهش	تعداد نمونه باقی‌مانده	تعداد ویژگی باقی‌مانده	دقت	زمان اجرا (ثانیه)	RMSE	Kappa
Original data set	0 %	50000	784	86.53	-	0.1575	0.8503
ReliefF	50 %	50000	392	86.73	34975	0.1567	0.6023
All k-NN	2.31 %	48843	784	87.80	61320	0.1499	0.8643
All k-NN+ ReliefF	51.15 %	48843	392	88.45	94623	0.1463	0.4802
IFSB-ReliefF with Jaccard Index	75 %	25000	392	85.07	8671	0.1657	0.8341
IFSB-ReliefF with Jaccard index	91 %	15000	235	80.49	8649	0.1898	0.7962
IFSB-ReliefF with Manhattan distance	75%	25000	392	85.38	8237	0.1634	0.8374
IFSB-ReliefF with Manhattan distance	91 %	15000	235	84.27	8313	0.1708	0.8252

(جدول-۴): نتایج اعمال الگوریتم‌های کاهش داده بر مجموعه‌داده German Credit

(Table-4): The results of applying data reduction algorithms on German Credit data set

الگوریتم کاهش داده	نرخ کاهش	تعداد نمونه‌های باقی‌مانده	تعداد ویژگی‌های باقی‌مانده	دقت	زمان اجرا (ثانیه)	RMSE	Kappa
Original data set	0%	1000	20	72.65	-	0.4836	0.2687
ReliefF	50%	1000	10	74.70	1	0.4221	0.284
All k-NN	30.2%	698	20	90.29	2	0.298	0.3737
All k-NN+ ReliefF	65.1%	698	10	87.34	2.8	0.3329	0
IFSB-ReliefF with Jaccard Index	75 %	500	10	71.76	44.7	0.4574	0.2793
IFSB-ReliefF with Jaccard index	89.5%	300	7	76.47	44.3	0.4031	0.4344
IFSB-ReliefF with Manhattan distance	75%	500	10	84.12	112.5	0.3639	0.5921
IFSB-ReliefF with Manhattan distance	89.5%	300	7	89.21	112.4	0.3067	0.7414

(جدول-۵): مقایسه نتایج به‌دست‌آمده به‌وسیلهٔ IFSB-ReliefF و روش‌های دیگر برای مجموعه‌داده German credit

(Table-5): Comparison between results obtained by IFSB-ReliefF and other methods for German credit

الگوریتم کاهش داده	تعداد نمونه‌ها	تعداد ویژگی‌ها	دقت	زمان اجرا(ثانیه)	Kappa
IFS-CoCo [1]	-	-	71.8	555.26	0.2335
GA:FSIS [30]	55	8	69.2	-	-
GA:ISFS [30]	210	6	67.2	-	-
IFSB-ReliefF with Jaccard Index	500	10	66.2	44.7	0.178
IFSB-ReliefF with Jaccard index	300	7	71	44.3	0.3369
IFSB-ReliefF with Manhattan distance	500	10	66.6	112.5	0.194
IFSB-ReliefF with Manhattan distance	300	7	68.67	112.4	0.2295

جدول (۶) نتایج را برای مجموعه‌داده Census نشان می‌دهد. اگرچه ReliefF برای کاهش داده زمان کمی صرف کرده و دقت نتایج آن خوب است، اما نرخ کاهش آن از الگوریتم پیشنهادی کمتر بوده و فقط ویژگی‌ها را کاهش می‌دهد. هم‌چنین، All k-NN با زمان اجرای کمتری نسبت به IFSB-ReliefF به نتایجی با دقت بالا (حتی دقیق‌تر از

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

مجموعه‌داده اولیه) می‌رسد، اما نرخ کاهش ان بسیار پایین بوده و درعمل به هدف کاهش داده‌ها دست نمی‌یابد. چنان‌که در جدول دیده می‌شود، الگوریتم IFSB-ReliefF با فاصله منهتن به نتایجی با دقت و بالاتر از همه، در عین اینکه نرخ کاهش داده نیز از همه بیشتر است، دست یافته است. همچنین نرخ خطای RMSE از همه کمتر و شاخص Kappa نیز بیشتر از همه است. یعنی الگوریتم با ثبات بیشتری عمل کرده است. البته این الگوریتم زمان بیشتری برای اجرا صرف می‌کند که دلیل آن وجود داده‌های غیر عددی و مقادیر مفقود و زمان‌بربودن محاسبه شباهت بین این مقادیر است. در مورد این مجموعه‌داده نیز، شاخص ژاکارد زمان کمتری برای کاهش داده مصرف کرده است، اما در شاخص‌های رده‌بندی عملکرد ضعیف‌تری از فاصله منهتن داشته است.

(جدول-۶): نتایج اعمال الگوریتم‌های کاهش داده بر مجموعه‌داده Census

(Table-6): The results of applying data reduction algorithms on Census dataset.

الگوریتم کاهش داده	نرخ کاهش	تعداد نمونه‌های باقی‌مانده	تعداد ویژگی‌های باقی‌مانده	دقت	زمان اجرا(ثانیه)	RMSE	Kappa
Original data set	0 %	30000	41	95.47	-	0.1963	0.4604
ReliefF	52 %	30000	20	94.98	490	0.2037	0.3707
All k-NN	5.41 %	28377	41	99.23	2396	0.0835	0.7202
All k-NN+ ReliefF	53 %	28377	20	99.06	2832	0.0954	0.4302
IFSB-ReliefF with Jaccard Index	75 %	15000	21	94.13	7098	0.2245	0.239
IFSB-ReliefF with Jaccard index	90.48 %	9000	13	94.50	7017	0.2131	0.2375
IFSB-ReliefF with Manhattan distance	75 %	15000	21	99.05	11529	0.0946	0.9375
IFSB-ReliefF with Manhattan distance	90.48 %	9000	13	99.31	11201	0.0788	0.9084

(جدول-۷): نتایج اعمال الگوریتم‌های کاهش داده بر مجموعه‌داده Balance-scale

(Table-7): The results of applying data reduction algorithms on Balance-scale data set

الگوریتم کاهش داده	نرخ کاهش	تعداد نمونه‌های باقی‌مانده	تعداد ویژگی‌های باقی‌مانده	دقت	زمان اجرا(ثانیه)	RMSE	Kappa
Original data set	0 %	625	4	74.057	-	0.3651	0.5495
ReliefF	50%	625	2	64.62	0.9	0.3978	0.3511
All k-NN	9.6%	565	4	87.5	1	0.3373	0.7496
All k-NN+ ReliefF	54.8%	565	2	73.44	1.7	0.4317	0.4701
IFSB-ReliefF with Jaccard Index	74.96%	313	2	56.60	5.15	0.4269	0.2163
IFSB-ReliefF with Jaccard index	92.52%	187	1	54.69	5.45	0.4128	0.2149
IFSB-ReliefF with Manhattan distance	74.96%	313	2	78.302	1.6	0.345	0.5972
IFSB-ReliefF with Manhattan distance	92.52%	187	1	78.125	2.2	0.3485	0.6007

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

(جدول-۸): مقایسه نتایج به‌دست‌آمده توسط IFSB-ReliefF و روش‌های دیگر برای مجموعه‌داده Balance-scale

(Table-8): Comparison between results obtained by IFSB-ReliefF and other methods for Balance-scale

الگوریتم کاهش داده	تعداد نمونه‌ها	تعداد ویژگی‌ها	دقت	زمان اجرا(ثانیه)	Kappa
IFS-CoCo [1]	–	–	84.95	69.58	0.7578
GA:FSIS [30]	68	3	72.70	–	–
GA:ISFS [30]	28	2	62.38	–	–
IFSB-ReliefF with Jaccard Index	313	2	53.35	5.15	0.1361
IFSB-ReliefF with Jaccard index	187	1	61.49	5.45	0.2871
IFSB-ReliefF with Manhattan distance	313	2	80.511	1.6	0.6391
IFSB-ReliefF with Manhattan distance	187	1	77.54	2.2	0.5842

در جدول (۷) نتایج مجموعه‌داده Balance-scale ارائه شده است. همان‌طور که مشاهده می‌شود، بالاترین درصد دقت مربوط به روش All k-NN است؛ اما این روش نرخ کاهش کمی دارد. همچنین این مجموعه‌داده یک مجموعه نامتوازن است و همان‌گونه که در بخش ۳ گفته شد، All k-NN مانند بسیاری از روش‌های انتخاب نمونه، این مسأله را در نظر نمی‌گیرد. برای مثال با اجرای All k-NN روی مجموعه‌داده Balance-scale، یکی از رده‌ها که تعداد نمونه‌های کمتری داشت، به‌طور کلی از مجموعه‌داده حذف شد. پس از All k-NN، روش پیشنهادی این مقاله با تابع فاصله منهتن بیشترین دقت را کسب کرده است که با توجه به نرخ کاهش زیاد و زمان اجرای کم و همچنین، حفظ نسبت نمونه‌های هر رده، می‌تواند روش مناسبی برای کاهش این مجموعه داده محسوب شود.

در جدول (۸) نتایج به‌دست‌آمده توسط IFSB-ReliefF با نتایج مقالات [1] و [30] مقایسه شده است. همان‌گونه که ملاحظه می‌شود، دقت به‌دست‌آمده به‌وسیلهٔ IFSB-ReliefF از دقت روش مقاله [30] بهتر است. همچنین اگرچه دقت روش مقاله [1] بهتر از IFSB-ReliefF است، اما این روش زمان زیادی صرف کرده است. البته همان‌گونه که پیش‌تر بیان شد، سخت‌افزارهای مورد استفاده و پارامترها برای اجرای الگوریتم‌ها در این مقاله‌ها بیان نشده است.

۵- بحث و یافته‌های پژوهش

در این بخش به‌اختصار نتایج روش‌های مختلف با هم مقایسه شده و نتایجی از یافته‌ها مستخرج می‌شود.

در جدول (۹) میانگین نتایج مربوط به روش‌های مختلف مشاهده می‌شود.

با توجه به مقادیری که در این جدول و سایر نتایج مشاهده می‌شود، می‌توان نکات و یافته‌های این مقاله را به‌صورت زیر خلاصه کرد:

۱. اگرچه الگوریتم All k-NN بالاترین میانگین دقت را به‌دست آورده است، اما کمترین نرخ کاهش را دارد و سبب می‌شود که برای کاهش داده در مجموعه‌داده‌های بزرگ مفید نباشد؛ علاوه‌براین، همان‌گونه که در نتایج مجموعه Balane-scale گفته شد، این الگوریتم برای مجموعه‌های نامتوازن مناسب نیست و ممکن است که نمونه‌های یک رده به‌طور کلی حذف شود.

۲. اگرچه می‌توان نرخ کاهش را در الگوریتم ReliefF تعیین کرد، اما این الگوریتم چند ضعف در زمینه کاهش داده دارد، نخست این‌که فقط کاهش ویژگی را انجام می‌دهد و در مجموعه‌داده‌هایی که تعداد نمونه زیاد و تعداد ویژگی کمی دارند، چندان مناسب نیست. دوم این‌که شاخص Kappa این الگوریتم پایین و دلیل آن انتخاب تصادفی برخی نمونه‌ها برای اجرای الگوریتم است که نتایج را بی ثبات می‌کند.

۳. ترکیب الگوریتم‌های All k-NN و ReliefF، علاوه‌بر تمام معایب گفته‌شده، سبب افزایش بسیار زیاد زمان اجرا می‌شود و مقدار شاخص Kappa نیز کاهش می‌یابد.

۴. استفاده از شاخص ژاکارد سبب کاهش زمان اجرای الگوریتم پیشنهادی شده و کمترین زمان اجرا را دارد. البته کمترین میزان دقت در این شاخص، به‌دست آمده است. اما با داشتن مزایایی مثل مدت زمان اجرا، قابلیت اجرای موازی، حفظ حالت اولیه داده‌ها و نرخ کاهش بالا، برای مجموعه‌داده‌های بزرگ و به‌ویژه نامتوازن می‌تواند یک انتخاب مناسب باشد.

۵. درنهایت، باثبات‌ترین و دقیق‌ترین نتایج توسط الگوریتم IFSB-ReliefF با فاصله منهتن کسب شده است. البته زمان اجرای این روش، به‌ویژه در مجموعه‌داده‌های غیرعددی، بیشتر از شاخص ژاکارد است؛ بنابراین می‌توان در مجموعه‌داده‌های غیرعددی از روش شاخص ژاکارد و در داده‌های عددی از فاصله منهتن استفاده کرد.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

(جدول-۹): میانگین نتایج مربوط به الگوریتم‌های مختلف

(Table-9): The average of results for various algorithms

الگوریتم کاهش داده	نرخ کاهش	دقت	زمان اجرا(ثانیه)	RMSE	Kappa
Original data set	0%	82.18	-	0.3006	0.5322
ReliefF	51%	80.26	8867	0.2951	0.4020
All k-NN	12%	91.21	15930	0.2172	0.6770
All k-NN+ ReliefF	56%	87.07	24365	0.2516	0.3451
IFSB-ReliefF with Jaccard Index	75%	76.89	3946	0.3186	0.3922
IFSB-ReliefF with Jaccard index	91%	76.54	3921	0.3047	0.4208
IFSB-ReliefF with Manhattan distance	75%	86.71	4952	0.2417	0.7411
IFSB-ReliefF with Manhattan distance	91%	87.73	4889	0.2262	0.7689

۶- جمع‌بندی

رشد بی‌رویه داده‌ها و وجود داده‌های تکراری و دارای نوفه، نه‌تنها نیاز به دستگاه‌های ذخیره‌سازی عظیم دارد، بلکه دقت الگوریتم یادگیری ماشین را کاهش می‌دهد؛ بنابراین به روش‌هایی برای کاهش داده‌ها و استخراج اطلاعات مفید نیاز است. در این مقاله، الگوریتم انتخاب ویژگی ReliefF به روشی متفاوت استفاده می‌شود. در الگوریتم IFSB-ReliefF، مشابهت نمونه‌ها اندازه‌گیری شده و نزدیک‌ترین نمونه‌ها به هم تعیین می‌شوند. بر اساس مشابهت محاسبه‌شده، نمونه‌های خیلی شبیه به هم حذف می‌شوند. هم‌چنین، بر اساس همین مجموعه همسایگی، وزن هر ویژگی محاسبه و ویژگی‌ها با وزن کمتر حذف می‌شوند. نتایج نشان می‌دهند که اجرای این الگوریتم داده‌ها را در زمانی مناسب و با نرخ کاهش بالایی کاهش می‌دهد؛ علاوه‌براین، IFSB-ReliefF می‌تواند دقت رده‌بندی را در محدوده مجموعه‌داده اولیه حفظ کرده و در مواردی حتی بهبود بخشد. هم‌چنین، این الگوریتم با درنظرگرفتن احتمال پیشین هر رده، نسبت بین نمونه‌ها در مجموعه‌داده اولیه از بین نمی‌رود. به‌دلیل این‌که دستورها در حلقه اصلی الگوریتم در تکرارهای مختلف از هم مستقل‌اند، بنابراین می‌توان آن را با یک سامانه چندپردازنده به‌صورت موازی اجرا کرد. در پژوهش‌های بعدی قصد داریم این الگوریتم را با توابع ارزیابی متفاوت و روی مجموعه‌داده‌های مختلف آزمایش و هم‌چنین نتایج حاصل از سایر الگوریتم‌های را با آن‌ها مقایسه کنیم.

۷- مراجع 7- References

[1] J. Derrac, S. Garcia and F. Herrera, "IFS-CoCo: Instance and feature selection based on cooperative coevolution with nearest neighbor rule.," *Pattern Recognition* , vol. 43, no. 6, pp. 2082-2105, 2010.

[2] H. Liu and L. Yu, "Toward Integrating Feature Selection Algorithms for Classification and Clustering," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 4, pp. 491-502, 2005.

[3] V. Bolón-Canedo, N. Sánchez-Maroño and A. Alonso-Betanzos, "Recent advances and emerging challenges of feature selection in the context of big data," *Knowledge-Based Systems*, vol. 86, pp. 33-45, 2015.

[4] L. C. Molina, L. Belanche and À. Nebot, "Feature Selection Algorithms: A Survey and Experimental Evaluation," *in IEEE International Conference on Data Mining*, 2002.

[۵] ج .پورامینی ,ب .مینایی بیدگلی و م .اسماعیلی , "یک روش جدید انتخاب ویژگی یک‌طرفه در دسته‌بندی داده‌های متنی نامتوازن، "پردازش

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

methods based on information entropy," *in IEEE International Joint Conference on Neural Networks*, 2004.

[20] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16-28, 2014.

[21] I. Kononenko, E. Šimec and M. Robnik-Šikonja, "Overcoming the myopia of inductive learning algorithms with RELIEFF," *Applied Intelligence*, vol. 7, no. 1, pp. 39-55, 1997.

[22] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, 1993.

[23] L. A. R. Kenji Kira, "The feature selection problem: Traditional methods and a new algorithm," *AAAI*, vol. 2, pp. 129-134, 1992.

[24] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Machine learning* , vol. 53, no. (1-2), pp. 23-69, 2003.

[25] H. Liu and H. Motoda, Computational methods of feature selection, CRC Press, 2007.

[26] K. Yu, X. Xu, M. Ester and H.-P. Kriegel, "Feature weighting and instance selection for collaborative filtering: An information-theoretic approach," *Knowledge and Information Systems,* vol. 5, no. 2, pp. 201-224, 2003.

[27] T. Chen, X. Zhang, S. Jin and O. Kim, "Efficient classification using parallel and scalable compressed model andits application on intrusion detection," *Expert Systems with Applications*, vol. 41, pp. 5972–5983, 2014.

[28] W. T, Hadoop, The Definitive Guide, O'Reilly Media, Inc., 2012.

[29] P. Perner, "Prototype-based classification," *Applied Intelligence*, vol. 28, no. 3, pp. 238-246, 2008.

[30] C.-F. Tsai, W. Eberle and C.-Y. Chu, "Genetic algorithms in feature and instance selection," *Knowledge-Based Systems*, vol. 39, p. 240–247, 2013.

[31] F. Dimitris, D. Meretakis and L. Spiros, "Integrating Feature and Instance Selection for text classification," In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 2002.

[32] H. Ahn and K.-j. Kim, "Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach," *Applied Soft Computing*, vol. 9, no. 2, pp. 599-607, 2009.

[33] Z. Abbasi and M. Rahmani, "An Instance Selection Algorithm Based on ReliefF," *International Journal on Artificial Intelligence Tools*, vol. 28, no. 1, p. 1950001, 2019.

[34] I. Tomek, "An experiment with the edited nearest-neighbor rule," *IEEE Transactions on*

[5] علائم و داده‌ها ,۱۶(۱)، ۲۱-۴۰، ۱۳۹۸

J. Pouramini, B.Minaei-Bidgoli, M.Esmaeili, "A Novel One Sided Feature Selection Method for Imbalanced Text Classification", *JSDP*, 2019, vol. 16 (1), pp.21-40.

[6] P. S. Bradley, U. M. Fayyad and C. Reina, "Scaling clustering algorithms to large databases," in Proceedings of the Fourth International Conference on Knowledge Discovery & Data Mining, New York, 1998.

[7] H. Liu, H. Motoda and L. Yu, "A selective sampling approach to active feature selection," *Artificial Intelligence*, vol. 159, pp. 49-74, 2004.

[8] W. Cochran, Sampling Techniques, New York: Wiley, 1977.

[9] H. Liu and H. Motoda, Instance Selection and Construction for Data Mining, Boston,MA: Kluwer Academic, 2001.

[10] S. Garcı´a, J. Derrac, J. R. Cano and F. Herrera, "Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study," *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 34, no. 3, pp. 417-435, MARCH 2012.

[11] S. Garc´ıa, J. Derrac, J. R. Cano and F. Herrera, "Prototype Selection for Nearest Neighbor Classification: Survey of Methods," *IEEE Transactions On Pattern Analysis And Machine Intelligence.*, vol. 34, no. 3, pp. 417-435, 2012.

[12] F. H. M. L. Jose Ramon Cano, "On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining," *Applied Soft Computing* , vol. 6, pp. 323–332, 2006.

[13] S. d. Río, V. Lopez, J. M. Benítez and F. Herrera, "On the use of MapReduce for imbalanced big data using Random Forest," *Information Sciences* , vol. 285, pp. 112-137, 2014.

[14] d. Wilson and t. r. Martinez, "Reduction techniques for instance-based learning algorithms," *Machine learning*, vol. 38, no. 3, pp. 257-286, 2000.

[15] H. Liu and H. Motoda, "On issues of instance selection," *Data Mining and Knowledge Discovery*, vol. 6, no. 2, pp. 115-130, 2002.

[16] D. R. Wilson and M. Tony R, "Instance pruning techniques," *ICML*, vol. 97, pp. 403-411, 1997.

[17] p. Jaccard, "Étude comparative de la distribution florale dans une portion des Alpes et des Jura," *Bulletin de la Société Vaudoise des Sciences Naturelles*, vol. 37, p. 547–579, 1901.

[18] G. H. J. Ron Kohavi, "Wrappers for feature subset selection," Artificial intelligence, vol. 97, no. 1, pp. 273-324, 1997.

[19] W. Duch, T. Wieczorek, J. Biesiada and M. Blachnik, "Comparison of feature ranking

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

Systems, Man, and Cybernetics, vol. 6, pp. 448–452, 1976.

[35] J. R. Quinlan, "Improved use of continuous attributes in c4.5.," *Journal of Artificial Intelligence Research*, vol. 4, pp. 77-90, 1996.

[36] I. Triguero, D. Peralta, J. Bacardit, S. García and F. Herrera, "MRPR: A MapReduce solution for prototype reduction in big data classification," *Neurocomputing*, vol. 150, pp. 331–345, 2015.

[37] R. Hyndman and K. Anne B., "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, no. 4, pp. 679–688, 2006.



زینب عباسی. دکترای خود را در رشته مهندسی نرم‌افزار در دانشگاه اراک در سال ۱۳۹۸ دریافت کرده است. حوزه پژوهشی ایشان، داده‌کاوی و الگوریتم‌های کاهش داده است.

نشانی رایانامه ایشان عبارت است از:

Zabasi@gmail.com



محسن رحمانی. مدرک دکترای خود را در سال ۱۳۸۷ از دانشگاه علم و صنعت ایران دریافت کردند. ایشان هم‌اکنون عضو هیأت علمی دانشگاه اراک هستند. تخصص ایشان پردازش سیگنال‌های صوتی و هوش مصنوعی است.

نشانی رایانامه ایشان عبارت است از:

m-rahmani@araku.ac.ir



حسین غفاریان. دکترای خود را از دانشگاه علم و صنعت ایران در سال ۱۳۹۲ دریافت کردند و در حال حاضر عضو هیأت علمی دانشگاه اراک هستند. حوزه تخصصی فعالیت ایشان داده‌کاوی و الگوریتم‌های تکاملی است.

نشانی رایانامه ایشان عبارت است از:

H-ghaffarian@araku.ac.ir

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]

[DOI: 10.29252/jsdp.17.4.49]

[DOI: 10.29252/jsdp.17.4.49] [Downloaded from jsdp.rcisp.ac.ir on 2026-11-14]